IN THE CLAIMS

This listing of claims replaces all prior listings:

1. (Previously Presented) A method in a data processing system for developing a data flow program comprising code segments that operate on data in memory, the method comprising the steps of:

dividing the memory into blocks;

assigning at least a portion of the data and at least one code segment to each block;

storing data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data read or written by the code segment;

determining whether dependencies exist among the blocks such that a first block depends on data assigned to a second block using the read and write identifiers; and

displaying a graph comprising the blocks and the determined dependencies.

2. (Original) A method according to claim 1, wherein the step of displaying comprises the step of displaying a graph comprising nodes assigned to the blocks and dependency arcs representing the determined dependencies.

3. (Original) A method according to claim 2, wherein the step of displaying further comprises the step of presenting the dependency arcs using a satisfied dependency visualization when the determined dependency is satisfied, and presenting the dependency arcs using an unsatisfied dependency visualization when the determined dependency is unsatisfied.

4. (Original) A method according to claim 2, further comprising the steps of:

receiving a node selection specifying a selected one of the nodes;

determining unmet dependencies for the selected node; and

displaying in a visually distinctive manner the unmet dependencies in the graph.

5. (Original) A method according to claim 2, further comprising the steps of:

providing for execution of the code segments using threads;

receiving a thread selection specifying at least one of the threads; and

displaying nodes executed by the at least one thread.

6. (Currently Amended) A method according to claim 2 ~~1~~, wherein the nodes include executed nodes and unexecuted nodes, and wherein the step of displaying further comprises the step of displaying the unexecuted nodes using an unexecuted visualization and the executed nodes using an executed visualization.

7. (Original) A method according to claim 1, wherein the data includes a data structure, and wherein the step of displaying further comprises the step of:

facilitating visualization of at least a portion of the data structure accessed by at least one of the code segments by graphically presenting at least a portion of the data structure and accentuating the portion of the data structure accessed by the at least one code segment.

8. (Previously Presented) A method in a data processing system for developing a data flow program comprising code segments distributed between memory blocks, the method comprising the steps of:

representing the data flow program as a graph comprising nodes and node dependencies between the nodes; and

displaying the graph to facilitate visualization of the data flow program,

wherein the node dependencies between nodes are determined based on data read and data write identifiers for code segments associates with the respective nodes, the data read and data write identifiers identifying at least a portion of data read or written by the respective code segment.

9. (Original) A method according to claim 8, wherein the nodes include executed nodes and unexecuted nodes, and wherein the step of displaying comprises the step of displaying the unexecuted nodes with an unexecuted visualization and displaying the executed nodes with an executed visualization.

10. (Original) A method according to claim 9, wherein the nodes include executing nodes, and wherein the step of displaying comprises the step of displaying the executing nodes with an executing visualization.

11. (Original) A method according to claim 8, wherein the node dependencies include satisfied dependencies and unsatisfied dependencies, and wherein the step of displaying comprises the steps of displaying the unsatisfied dependencies using an unsatisfied dependency

visualization, and displaying the satisfied dependencies using a satisfied dependency visualization.

12. (Previously Presented) A computer-readable medium containing instructions that cause a data processing system to perform a method for developing a data flow program comprising code segments that operate on data in memory, the method comprising the steps of:

dividing the memory into blocks;

assigning at least a portion of the data and at least one code segment to each block;

storing data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data read or written by the code segment;

determining a dependency imparted by a first block depending on data assigned to a second block using the read and write identifiers; and

displaying a graph comprising the blocks and the determined dependency.

13. (Original) A computer-readable medium according to claim 12, wherein the step of displaying comprises the step of displaying a graph comprising nodes assigned to the blocks and a dependency arc representing the determined dependency.

14. (Previously Presented) A computer-readable medium according to claim 13, wherein the step of displaying further comprises the step of presenting the dependency arc using a satisfied dependency visualization when the determined dependency is satisfied, and presenting the dependency arc using an unsatisfied dependency visualization when the determined dependency is unsatisfied.

15. (Original) A computer-readable medium according to claim 13, further comprising the steps of:

receiving a node selection specifying a selected node;

determining unmet dependencies for the selected node; and

highlighting in the graph the unmet dependencies.

16. (Original) A computer-readable medium according to claim 13, further comprising the steps of:

providing for execution of the code segments using threads;

receiving a thread selection specifying at least one of the threads; and

displaying nodes executed by the at least one thread.

17. (Previously Presented) A computer-readable medium according to claim 13, wherein the nodes include executed nodes and unexecuted nodes, and wherein the step of displaying further comprises the step of presenting the unexecuted nodes using an unexecuted visualization and the executed nodes using an executed visualization.

18. (Original) A computer-readable medium according to claim 12, wherein the data includes a data structure, and wherein the step of displaying further comprises the step of:

facilitating visualization of at least a portion of the data structure accessed by at least one of the code segments by graphically presenting at least a portion of the data structure and accentuating the portion of the data structure accessed by the at least one code segment.

19. (Original) A method in a data processing system for developing a data flow program comprising code segments that operate on data in a memory, the method comprising the steps of:

dividing into blocks the memory that stores the data;

for each block, assigning at least a portion of the data to the block and assigning at least one of the code segments to the block;

storing data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data read or written by the code segment;

determining whether dependencies exist among the blocks such that a first block depends on data assigned to a second block using the read and write identifiers;

generating a directed acyclic graph comprising nodes and arcs between the nodes by assigning the blocks to the nodes and by assigning the dependencies to the arcs;

displaying the directed acyclic graph;

initiating execution of the code segments;

while the code segments are executing,

determining which nodes in the graph are unexecuted nodes and which nodes in the graph are executed nodes; and

displaying the unexecuted nodes in a manner visually distinctive from the executed nodes.

20. (Previously Presented) A data processing system comprising:

a memory comprising a data flow program and a data flow development tool that associates data processed by the data flow program to blocks in the memory, associates code

segments of the data flow program to at least one of the blocks, stores data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data read or written by the code segment, determines dependencies between the blocks using the read and write identifiers, and displays a graph comprising nodes and arcs depicting the dependencies between the blocks; and

a processor that runs the data flow development tool.

21. (Original) The data processing system of claim 20, wherein the nodes comprise executed nodes and unexecuted nodes, and wherein the executed nodes are displayed using an executed node visualization and the unexecuted nodes are displayed using an unexecuted node visualization.

22. (Original) The data processing system of claim 20, wherein the arcs comprise satisfied dependency arcs and unsatisfied dependency arcs, and wherein the satisfied dependency arcs are displayed using a satisfied dependency visualization and the unsatisfied dependency arcs are displayed using an unsatisfied dependency visualization.

23. (Previously presented) A data processing system for developing a data flow program comprising code segments that operate on data in memory, the data processing system comprising:

means for apportioning a memory into regions and associating the data and the code segments with the regions;

means for storing data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data read or written by the code segment;

means for determining dependencies between the regions using the data read and write identifiers; and

means for displaying a graph of nodes that are assigned regions, and arcs depicting the dependencies between the regions.

24. (Previously Presented) A computer readable memory device encoded with a data structure accessed by a data flow development tool run by a processor in a system, the data structure comprising:

nodes assigned to data processed by a data flow program and to code segments of the data flow program;

data read and data write identifiers for each code segment, the data read and data write identifiers identifying at least a portion of the data read or written by the code segment; and

dependencies between nodes determined based on the data read and data write identifiers, wherein

the development tool accesses the data structure to provide a visualization of the data flow program.

25. (Original) A computer readable memory device according to claim 24, wherein the data structure further comprises:

a processed flag that indicates whether at least one of the nodes is executed or unexecuted.

26. (Original) A computer readable memory device according to claim 24, wherein the data structure further comprises:

a taken flag that indicates whether at least one of the nodes has been claimed by a thread.